memcpy_s() and memmove_s()

Daniel Plakosh, Software Engineering Institute [vita[1]]

2005-09-27; Updated 2008-10-06 L2 / D/P, L[2]

Substituting the `memcpy_s()` and `memmove_s()` functions for the `memcpy()` and `memmove()` functions can help guard against software vulnerabilities.

Development Context

Copying characters from one memory location to another.

Technology Context

C++, C, UNIX, Win32

Attacks

Attacker executes arbitrary code on machine with permissions of compromised process or changes the behavior of the program.

Risk

The `memcpy()` and `memmove()` functions are a source of buffer overflow vulnerabilities.

Description

The `memcpy_s()` and `memmove_s()` functions defined in ISO/IEC TR 24731 are similar to the corresponding less-secure `memcpy()` and `memmove()` functions but provide some additional safeguards. The secure versions of these functions add an additional argument that specifies the maximum size of the destination. The `memcpy_s()` and `memmove_s()` functions return zero if successful. A nonzero value is returned if either the source or destination pointer is NULL, if the specified number of characters to copy/move is greater than the maximum size of the destination buffer, or the number of characters to copy/move or the maximum size of the destination buffer is greater than RSIZE_MAX.[13]

References

[ISO/IEC 99]	ISO/IEC. *ISO/IEC 9899 Second edition 1999-12-01 Programming languages — C*. International Organization for Standardization, 1999.
[ISO/IEC 05]	ISO/IEC. *ISO/IEC TR 24731 Extensions to the C library -- Part 1: Bounds-checking interfaces*. International Organization for Standardization, 2005.

1. http://buildsecurityin.us-cert.gov/bsi/about_us/authors/268-BSI.html (Plakosh, Daniel)
13. The `RSIZE_MAX` is used to limit the size of objects passed to functions that have parameters of type `rsize_t` . Extremely large object sizes are frequently a sign that an object's size was calculated incorrectly. For example, negative numbers appear as very large positive numbers when converted to an unsigned type like `size_t`. Also, some implementations do not support objects as large as the maximum value that can be represented by type `size_t` . As a result, it is sometimes beneficial to restrict the range of object sizes to detect potential vulnerabilities.